Lez03b - Elis - IT Architect

Java Applet

Ing. Alberto Bucciero

Applet Java

- Le applet sono applicazioni sviluppate come normali programmi e compilate per essere eseguite solo tramite un apposito modulo software integrato nei browser Internet.
- Per poter eseguire la stessa applet su qualunque piattaforma, è sufficiente la presenza all'interno dei browser dell'interprete Java (JVM).
- Le applet sono applicazioni eseguite sul computer dell'utente.
- Questo rende la loro esecuzione sufficientemente veloce in quanto viene evitato traffico di dati con il server.

College
elis SCHOOL

Applet Java (cont.)

- Un'applet è un normale programma Java, con la stessa sintassi e le stesse parole chiave.
- Una volta sviluppata, l'applet deve essere richiamata in una pagina HTML e visualizzata mediante :
  - un browser (Internet Explorer, Firefox, ecc.)

    oppure
  - AppletViewer (letteralmente "visualizzatore di applet"), una applicazione apposita fornita con il JDK
  - Negli IDE come Eclipse o Netbeans

College
elis SCHOOL

Esempio banale

```
/*<APPLET CODE="CiaoAtutti.class" WIDTH=150
HEIGHT=25></APPLET>*/
import java.applet.*;
import java.awt.*;
public class CiaoATutti extends Applet {
      public void paint(Graphics g) {
            g.drawString("Ciao a tutti!", 5, 25);
      }
}
```


College

elis SCHOOL

Per scrivere un'applet…

- Occorre estendere la classe

  java.applet.Applet

- Occorre sovrascrivere alcuni dei suoi metodi
  - Nell'esempio abbiamo sovrascritto il metodo paint che viene invocato quando l'applet viene mostrata sullo schermo

Per visualizzare un'applet

- Occorre compilare il file .java (come sempre)
- Occorre un browser web
  - In questo caso ci serve un file html
- Possiamo usare anche l'appletviewer
  - In questo caso possiamo inserire il codice html come commento all'interno del file .java (come nell'esempio)

College
elis SCHOOL

Esempio

- Compilazione:
  - javac CiaoATutti.java
- Esecuzione
  - appletviewer CiaoATutti.java
- ma anche
  - appletviewer CiaoATutti.html
- oppure anche
  - appletviewer CiaoATutti.txt
- All'appletviewer basta che sia un file di testo con scritto da qualche parte
  - `<APPLET CODE="CiaoAtutti.class" WIDTH=150 HEIGHT=25></APPLET>`

College
elis SCHOOL

Tag HTML e applet

- L'inserimento all'interno della pagina HTML di un'Applet avviene per mezzo del tag <APPLET> con il quale tra l'altro si specifica il nome della classe Java che costituisce l'Applet stessa.
- Attualmente tutti i browser incorporano una macchina virtuale e sono in grado di eseguire un'Applet.
- Purtroppo si riscontrano sostanziali differenze sia fra le varie marche di browser , queste differenti scelte politiche e tecnologiche delle aziende possono causare di fatto un comportamento diverso di un'Applet in esecuzione a seconda del browser utilizzato.
- Ad esempio, per quanto riguarda la gestione dell'interfaccia grafica, proprio per garantire la massima portabilità, si è spesso costretti a effettuare una scelta conservativa limitandosi all'utilizzo di AWT, a scapito delle più potenti Swing API

College
elis SCHOOL

Applet e sicurezza

- Al fine di garantire un elevato margine di sicurezza le Applet devono rispettare i vincoli imposti dal Security Manager della macchina virtuale del browser, in particolare un'Applet non può:
  - accedere in nessun modo al file system della macchina locale;
  - eseguire programmi locali;
  - scrivere sul server da cui proviene;
  - accedere ad host diversi da quelli di provenienza;
  - trovare informazioni riguardanti il computer locale, tranne la versione java in uso.

College
elis SCHOOL

Problema

- Le politiche di sicurezza via via più stringenti che hanno adottato i browser web hanno reso sempre più difficile l'uso di applet.
- Al momento le uniche due soluzioni tecnologiche per usare una tecnologia essenzialmente desktop all'interno di un browser sono:
  - Applet firmate
  - JWS (Java Web Start)

- Nota:
  - Si tenga presente che già con l'introduzione del JDK 1.1, e ancora di più con Java 2, sono state messe a disposizione del programmatore delle tecniche avanzate per modificare la politica restrittiva del security manager tramite la firma delle Applet.

College
elis SCHOOL

Differenze tra Applet e applicazioni

- A causa del differente ambiente di esecuzione — la Virtual Machine del browser internet — le Applet presentano sostanziali differenze rispetto alle applicazioni Java standalone.
- Per prima cosa, per creare un'Applet, si deve necessariamente estendere la classe Applet (o JApplet nel caso si tratti di un'Applet che usa la libreria grafica Swing), mentre un'applicazione standalone può essere dichiarata indipendentemente da altre classi.
- Per l'Applet l'inizializzazione avviene all'interno del metodo init(), dove tra le altre cose è possibile ricavare i parametri passati dall'esterno

College
elis SCHOOL

Il tag <APPLET>

- Analizziamo ora alcuni attributi del tag <APPLET>:
  - WIDTH ed HEIGHT: questi attributi sono indispensabili e determinano larghezza e altezza dell'applet all'interno della pagina Web espressi in numero di pixel.
  - ALIGN: specifica l'allineamento dell'applet . Alcuni suoi valori (LEFT e RIGHT) fanno in modo che l'applet sia un blocco fisso attorno al quale scorra il testo, altri valori fanno in modo che l'applet sia un elemento in linea, ossia mobile all'interno di una riga di testo (BOTTOM= il bordo inferiore dell'applet combacia con il bordo inferiore della riga corrente, TOP, MIDDLE, …)
  - CODE: attributo indispensabile. Fornisce il nome del file compilato (.class) dell'applet. Può essere una directory locale o un url della rete.

HTML+Applet

- Una volta sviluppata, l'applet deve essere richiamata in una pagina HTML, tramite il tag <APPLET>:

```
<HTML>
        <HEAD>
                <TITLE>Un'applet di prova</ TITLE >
        </ HEAD >
        <BODY>
                <H1 ALIGN="center">La nostra prima applet</H1>
                <P ALIGN="center">
                        <APPLET CODE="Prima.class" WIDTH="400" HEIGHT="150">
                        </APPLET>
                </P>
        </BODY >
</ HTML >
```


College

elis SCHOOL

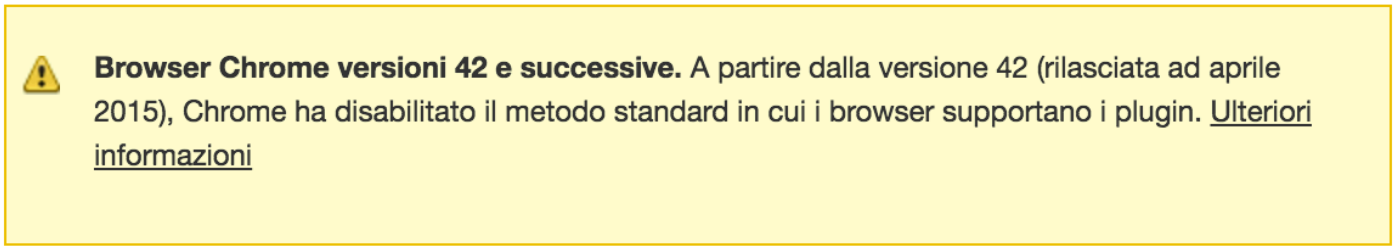
Come si abilita Java nel browser Web?

- **Internet Explorer**
  - Fate clic su **Strumenti**, quindi selezionate **Opzioni Internet**
  - Selezionate la scheda **Sicurezza**, quindi fate clic sul pulsante **Livello personalizzato**
  - Scorrete la lista fino a **Esecuzione script delle applet Java**
  - **Accertatevi che il pulsante di scelta Attiva sia selezionato**
  - Fate clic su **OK** per salvare le preferenze
- **Firefox**
  - Avviate il browser Firefox o riavviatelo se è già in esecuzione
  - Dal menu di Firefox, selezionate **Strumenti**, quindi fate clic sull'opzione **Componenti aggiuntivi**
  - Nella finestra Gestione componenti aggiuntivi, selezionate **Plugin**.
  - Fate clic sul plugin **Piattaforma Java(TM)** (Windows) o sul **plugin Applet Java** (Mac OS X) per selezionarlo.
  - Verificate che l'opzione selezionata sia **Richiesta attivazione** o **Attiva sempre** oppure, nelle versioni Firefox meno recenti, fare clic sul pulsante **Abilita** (se il testo del pulsante è **Disabilita**, vuol dire che Java è già abilitato).

Come si abilita Java nel browser Web? (cont.)

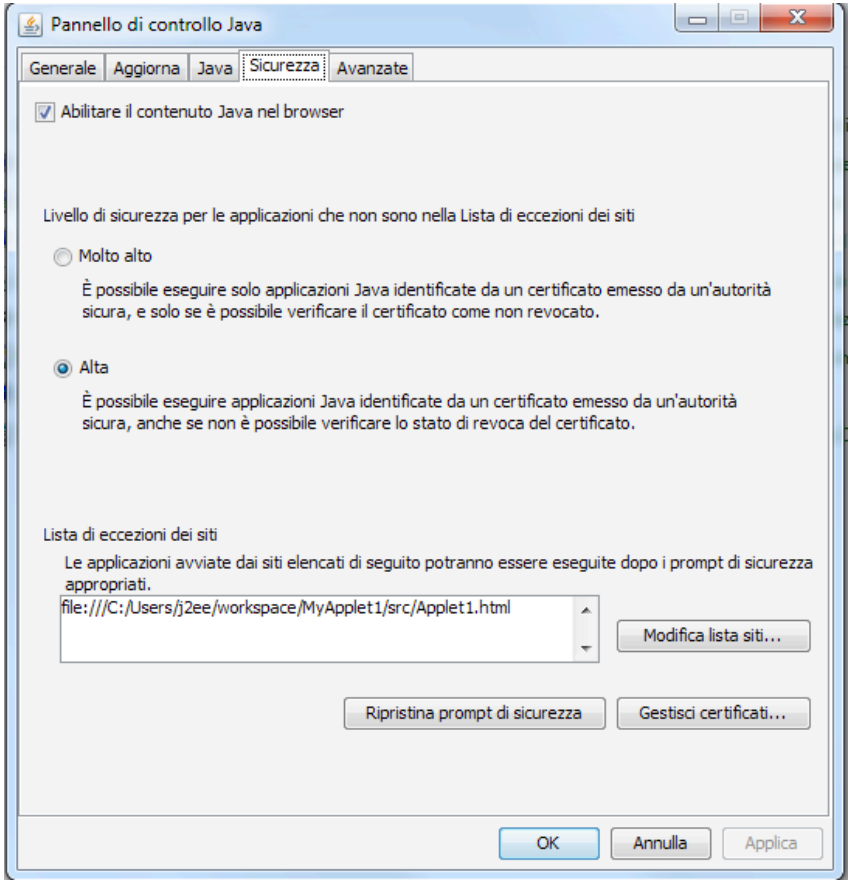
- **Safari**
  - Fate clic su Safari, quindi selezionate **Preferenze**.
  - Scegliete l'opzione **Sicurezza**.
  - Selezionate **Consenti plugin**, quindi fate clic su **Gestisci impostazioni del sito Web**
  - Fate clic sulla voce Java, selezionate un'opzione (Chiedi, Consenti o Sempre consentito) dall'elenco a discesa **Quando si visitano altri siti Web**
  - Fate clic su **Fine**, quindi chiudete la finestra Preferenze di Safari

Chrome



Come si abilita Java nel browser Web? (cont.)

- E’ anche necessario abilitare il contenuto Java nel browser da pannello di controllo Java



College
elis SCHOOL

Applet e classi Java (cont.)

- Ogni applet deve essere dichiarata come sottoclasse della classe java.applet.Applet:

import java.applet.Applet;

public class NomeApplet extends Applet { … }

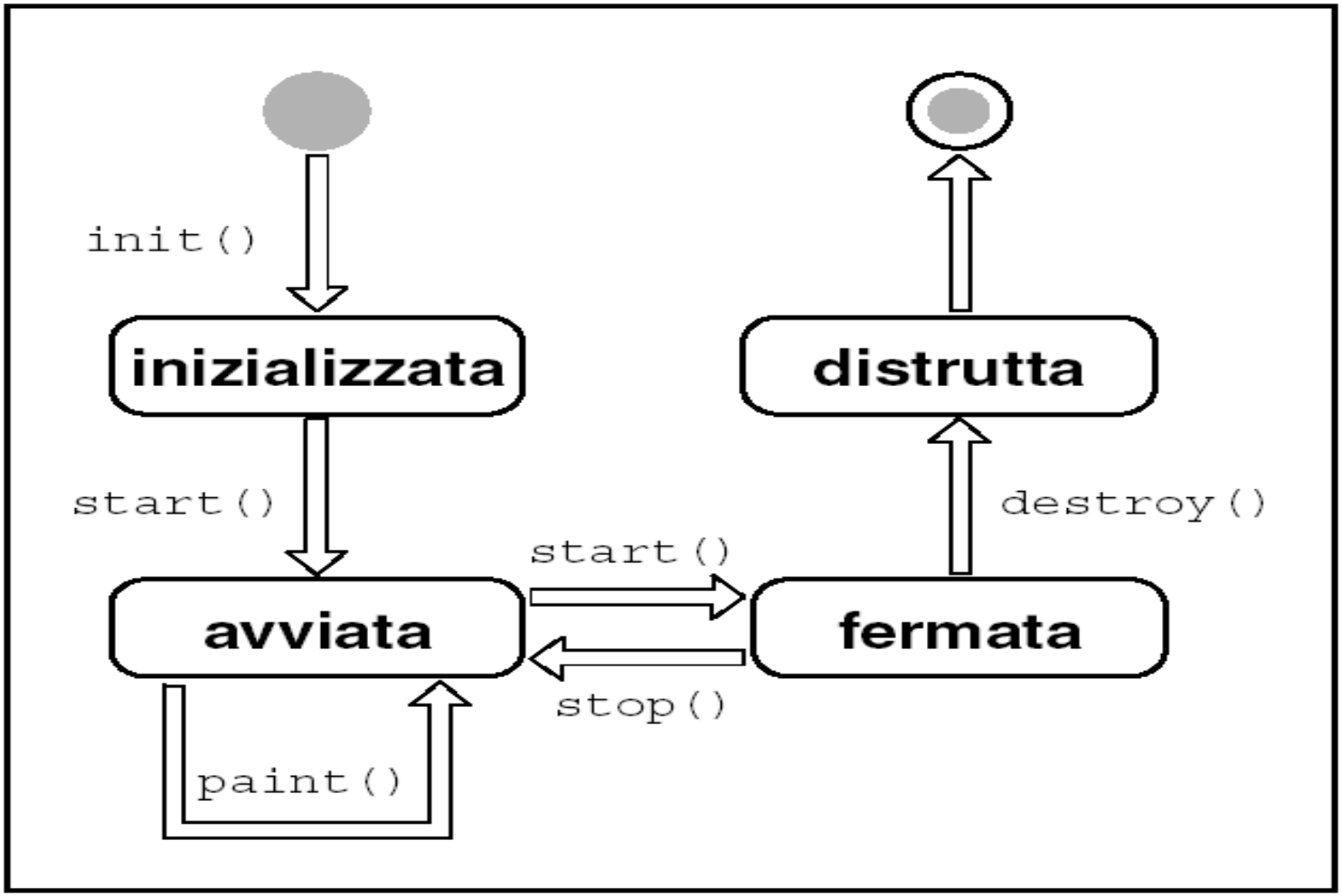
```
package it.elis.appletlesson;
import java.awt.*;
import java.applet.Applet;

public class myApplet extends Applet {
    Label t;
    public void init() {
          t=new Label("Ciao mondo!");
    add(t);
    }
    public void start() {
          setBackground(Color.green);
          setForeground(Color.yellow);
          repaint();
    }
}
```


College
elis SCHOOL

Applet e classi Java (cont.)

- In una applicazione Java, il metodo main rende l'applicazione eseguibile.
- Differentemente, nelle applet il metodo main non è ammesso, ma devono essere opportunamente utilizzati quattro metodi della classe Applet:
  - init()
  - start()
  - stop()
  - destroy()

Applet e classi Java (cont.)



College
elis SCHOOL

Il metodo init()

```
public void init() {
new Label("Ciao mondo!");
add(t);
}
```

- Serve per le inizializzazioni occorrenti per l'applet, analogamente ad un costruttore.
- Viene chiamato automaticamente dal sistema quando Java lancia l'applet la prima volta.
- Le applet possono avere costruttori propri, ma è opportuno eseguire tutte le inizializzazioni nel metodo init().

College
elis SCHOOL

Il metodo start()

```
public void start() {
  setBackground(Color.green);
  setForeground(Color.yellow);
  repaint();
}
```

- Viene chiamato automaticamente dopo il metodo init().
- Viene chiamato anche ogni volta che l'utente torna alla pagina contenente l'applet dopo aver visitato altre pagine. Quindi, può essere chiamato ripetutamente, a differenza di init(). Per questo, è preferibile inserire in init() anziché in start() le istruzioni che devono essere eseguite una sola volta.
- Il metodo start() è il punto in cui viene riavviata l'applet, ad esempio al ritorno alla pagina che la contiene.
- Se l'applet non contiene alcuna operazione che deve essere sospesa quando l'utente esce dalla pagina Web corrente, non occorre implementare start().

College
elis SCHOOL

Il metodo stop()

- Viene chiamato automaticamente dal sistema quando l'utente esce dalla pagina in cui si trova l'applet.
- Consente l'interruzione di un'attività che richiede molto tempo e che rallenta il sistema quando l'utente non presta attenzione all'applet.
- Se l'applet non esegue animazioni, non riproduce file audio e non esegue calcoli in un thread, di norma l'impiego di stop() non è necessario (infatti non era implementato nella classe CiaoMondo).

College
elis SCHOOL

Il metodo destroy() e altri metodi di applet

- Chiude l'applet e causa la pulizia della memoria ed il rilascio di tutte le risorse che erano state impegnate.
- Il suo impiego non è sempre richiesto (infatti non era implementato nella classe **CiaoMondo**).

```
public void start() {
  setBackground(Color.green);
  setForeground(Color.yellow);
  repaint();
}
```

- paint()e repaint() servono, rispettivamente, per la visualizzazione dell'applet e per l'aggiornamento della videata quando necessario.

College
elis SCHOOL

Applet, contenitori e il metodo add

- L'esecuzione di un'applet causa la creazione nella pagina HTML di un contenitore in cui l'applet stessa inserirà tutti gli elementi di interazione con il mondo esterno. Tali elementi, come vedremo, possono essere etichette, campi o aree di testo, pulsanti,…
- Per potere inserire uno di tali elementi nel contenitore si utilizza il metodo add della classe Applet.

```
public void init() {
   t=new Label("Ciao mondo!");
   add(t);
}
```

- Label è una classe del package java.awt che rappresenta brevi scritte

(etichette, appunto).

- Dopo aver dichiarato una variabile di tipo Label, il costruttore la inizializza ad una stringa (il parametro) ed il metodo add la inserisce nel contenitore.

College
elis SCHOOL

Gestori del Layout

- Invece di posizionare i componenti utilizzando un sistema di coordinate, in Java è possibile (e preferibile) associare al **Container** che contiene i vari componenti un oggetto **LayoutManager** che si occupa di *posizionare* e *dimensionare* i componenti secondo delle regole stabilite

FlowLayout

- Il gestore del layout FlowLayout è quello predefinito per i Panel e le Applet
- Posiziona gli elementi come se fossero le parole di un paragrafo con allineamento centrato: un flusso da sinistra verso destra
- È possibile modificare l'allineamento e lo spazio orizzontale e verticale tra gli elementi

College
elis SCHOOL

Esempio FlowLayout

```
import java.awt.*;
import java.applet.Applet;
public class ex01 extends Applet {
    public void init() {
        add(new Button("Ok"));
        add(new Button("Apri"));
        add(new Button("Chiudi"));
        }
    }
```

Come si cambia il gestore del layout

- La classe **Container** (e quindi tutte le sottoclassi) ha il metodo setLayout(LayoutManager m)
- Es: **a1.setLayout(new FlowLayout())** imposta un FlowLayout manager come gestore per il Container a1
- **a2.setLayout(new FlowLayout(FlowLayout.LEFT))** come prima, ma con allineamento a sinistra
- **a3.setLayout(new FlowLayout(FlowLayout.RIGHT,20,40))** come prima, ma con allineamento a destra, distanza orizzontale di 20 e distanza verticale di 40

College
its SCHOOL

BorderLayout

- È il gestore predefinito per i Frame
- Posiziona al massimo 5 elementi in 5 posizioni:
  - NORTH, EAST, SOUTH, WEST, CENTER
- I componenti vengono "stirati" per occupare tutto lo spazio a disposizione

Esempio BorderLayout


```
import java.awt.*;
import java.applet.Applet;
public class ex02 extends Applet {
        public void init() {
                this.setLayout(new BorderLayout());
                add(new Button("Ok"),BorderLayout.CENTER);
                add(new Button("Apri"),BorderLayout.EAST);
                add(new Button("Chiudi"),BorderLayout.NORTH); }
}
```

GridLayout

- Posiziona gli elementi in una griglia
- La dimensione della griglia viene definita con il costruttore GridLayout(int rows, int cols)
- I componenti vengono "stirati" per occupare tutto lo spazio a disposizione
- È possibile indicare nel costruttore anche lo spazio orizzontale e verticale (come in FlowLayout)

College
elis SCHOOL

Esempio GridLayout

```
import java.awt.*;
import java.applet.Applet;
public class ex03extends Applet {
      public void init() {
            this.setLayout(new
GridLayout(2,2,10,20));
            add(new Button("Ok"));
            add(new Button("Apri"));
            add(new Button("Chiudi"));
      }
}
```

Movimento del Mouse

- Un ascoltatore degli eventi relativi al movimento del mouse deve implementare l'interfaccia MouseMotionListener
- Ha due metodi:
  - public void mouseDragged(MouseEvent e)
  - public void mouseMoved(MouseEvent e)

College
elis SCHOOL

Eventi della tastiera

- L'interfaccia KeyListener definisce i metodi per gestire gli eventi della tastiera
  - public void keyTyped(KeyEvent e)
  - public void keyPressed(KeyEvent e)
  - public void keyReleased(KeyEvent e)

College
elis SCHOOL

Focus

- Un componente ha il focus quando è pronto per ricevere l'input (ad esempio quando si clicca su una casella di testo)
- L'interfaccia FocusListener definisce i metodi per gestire l'acquisto e la perdita del focus
  - void focusGained(FocusEvent e)
  - void focusLost(FocusEvent e)

College
elis SCHOOL

Interazione con un'applet

- L'applet che abbiamo appena visto scrive un messaggio in una finestra senza richiedere alcuna interazione con l'utente. Per poter interagire con un'applet (ad esempio, per comunicare dei dati) dobbiamo introdurre il concetto di evento.
- Un evento è un'azione che avviene su un certo oggetto o l'accadere di un errore.
- Esempi di eventi: fare clic su un pulsante, scegliere una voce tra le opzioni di un menu, trascinare un oggetto con il mouse, …
- In ogni caso l'evento viene comunicato a chi è incaricato di intraprendere le decisioni (e le azioni) opportune (event listener).

College
elis SCHOOL

Oggetti ed eventi

- Distinguiamo eventi di azione ed eventi di scorrimento.
- Le sorgenti di eventi di azione sono classi raccolte nel package java.awt. Le principali sono:
  - campi ed aree di testo (TextField e TextArea)
  - pulsanti (Button)
  - menu ed elenchi (Choice e List)
  - caselle di controllo (CheckBox)
- L' evento classico di scorrimento è relativo alla classe che implementa barre di scorrimento (ScrollBar).
- Altri eventi riguardano l'interazione mediante mouse.

College
elis SCHOOL

Un package di strumenti utili: awt

- Acronimo di Abstract Windowing Toolkit contiene un insieme di classi che permettono lo sviluppo di interfacce grafiche, utili per la gestione di finestre e loro componenti: menu, aree di testo, pulsanti, caselle di selezione,…
- Per il momento, abbiamo già incontrato una Label: la classe Label permette di visualizzare un testo per mezzo di un parametro passato al suo costruttore e quindi utilizzare il metodo add() per inserirlo nel contenitore corrente.
- Textfield è una classe di java.awt analoga a Label e definisce le caselle di testo, cioè aree di lunghezza fissa (determinata dai parametri al momento della inizializzazione) in cui è possibile visualizzare stringhe. Fra i metodi di Textfield troviamo:
  - void setEditable(boolean …): permette di definire il campo di testo “modificabile”, cioè, consente all’utente di scrivere nella casella
  - boolean isEditable():verifica se la casella è modificabile.

Un package di strumenti utili: awt (cont.)

- La classe Button è anche essa simile alla classe Label e definisce pulsanti. Differentemente da Label ha la caratteristica di essere reattiva: è possibile fare in modo che il programma esegua una determinata azione in risposta all'evento di pressione di un pulsante da parte dell'utente.
- Per reagire alla pressione di un pulsante, in java.awt è definita l'interfaccia actionListener che contiene il solo metodo actionPerformed che deve essere implementato dalla classe Button.

College
elis SCHOOL

Gestione di eventi di azione

- Per gestire gli eventi di azione in java esiste la apposita interfaccia java.awt.event. L'interfaccia event a sua volta è specializzata in ulteriori sottointerfacce a seconda degli eventi che si devono gestire
- Ad esempio tutti gli eventi di azione legati ai bottoni utilizzano il metodo actionPerformed(ActionEvent) di java.awt.event:

```
public void actionPerformed(ActionEvent ae) {
  //istruzioni di gestione dell'evento ae
}
```

- Il metodo andrà di volta in volta implementato per gestire l'evento di interesse.
- Tra i diversi metodi della classe ActionEvent ci sono getActionCommand() e getSource(), che permettono di rilevare l'oggetto che ha generato l'evento

Esempio

```
1  package it.elis.appletlesson;
2
3  import java.awt.*;
4  import java.applet.Applet;
5  import java.util.*;
6  import java.awt.event.*;
7
8  @SuppressWarnings("serial")
9  public class applet2 extends Applet implements ActionListener
10 {
11   Button P1, P2;
12   TextField testo;
13   public void init() {
14     P1=new Button("pari");
15     add(P1);
16     P1.addActionListener(this);
17     P2=new Button("dispari");
18     add(P2);
19     P2.addActionListener(this);
20     testo =new TextField();
21     testo.setColumns(30);
22     add(testo);
23   }
```

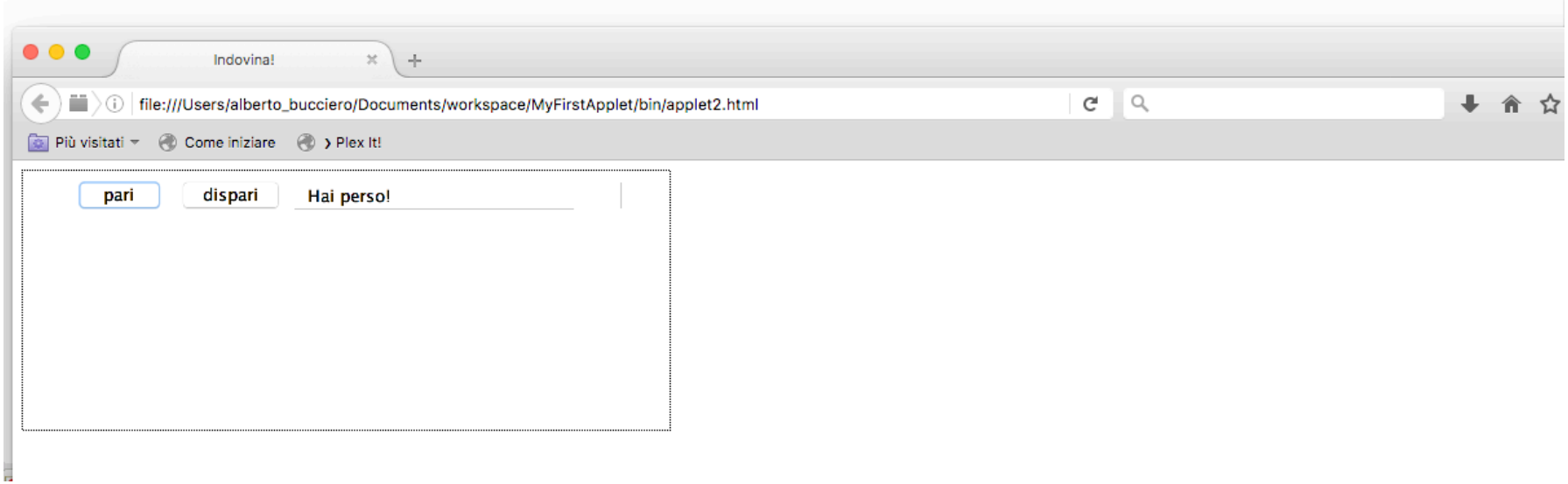
```
25 //implementiamo il metodo actionPerformed()
26 public void actionPerformed(ActionEvent ae) {
27   int i,a;
28   String valore;
29   Random r=new Random();
30   a=Math.abs(r.nextInt())%2;
31   valore=(String) ae.getActionCommand();
32   if ((a==0)&&(valore.equals("pari")))
33     testo.setText(" Hai vinto!");
34   else if ((a==1)&&(valore.equals("dispari")))
35    testo.setText(" Hai vinto!");
36   else testo.setText(" Hai perso!");
37  }
38 }
```


College

elis SCHOOL

Chiamare l'Applet da HTML

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4  <meta charset="UTF-8">
5  <title>Indovina!</title>
6  </head>
7  <body>
8      <applet code= "it/elis/appletlesson/applet2.class" height="200" width="500"> Pari o dispari?</applet>
9  </body>
10 </html>
```



Applet e Parametri

- Il file HTML può specificare parametri da passare all'applet, nella forma:

```
<APPLET CODE="Applet3.class"
WIDTH=500 HEIGHT=300>
<PARAM NAME="ascissa" VALUE="123">
<PARAM NAME="ordinata" VALUE="67">
...
</APPLET>
```

- L'applet può recuperarli con il metodo getParameter(nomeparametro), che restituisce una String

Applet e Parametri (cont.)

```
import java.applet.*;
import java.awt.*;
import javax.swing.*;
public class Applet4 extends JApplet {
    Font f = new Font("Times", Font.BOLD, 36);
    public void paint(Graphics g) {
          g.setFont(f);
          g.setColor(Color.red);
          g.drawString(getParameter("Frase"), 100, 50);
    }
}
```

```
Pagina html
<HTML><HEAD>
<TITLE> Applet Parametrica </TITLE>
</HEAD> <BODY>
<APPLET CODE="Applet4.class"
WIDTH=500 HEIGHT=100 >
<PARAM NAME="Frase" VALUE="Parametri passati!">
<!-- NOME e VALORE del parametro -->
</APPLET>
</BODY>
</HTML>
```


College
elis SCHOOL

Il codice HTML per l'Applet

tag	significato
`ALIGN` (opzionale)	Allineamento dell'Applet nella pagina HTML
`ALT` (opzionale)	Possibile messaggio di JVM non abilitata
`ARCHIVE` (opzionale)	Indica il file di archivio (estensione `.jar`) nel quale cercare le classi Java (file di estensione `.class`)
`CODEBASE` (opzionale)	Indica il percorso da webroot dove ricercare il file di estensione `.class` dell'applet
`CODE` (non opzionale)	Indica il file applet (estensione `.class`) da caricare
`NAME` (opzionale)	Assegna all'applet un nome referenziabile da JavaScript o usato nell'`appletContext` per la comunicazione fra applet nello stesso contesto
`HSPACE` (opzionale)	Definisce la spaziatura orizzontale.
`HEIGHT` (non opzionale)	Definisce l'altezza del canvas dell'applet che non può essere ridefinita a runtime. `Canvas` è una classe che implementa la classe astratta `Component` e che rappresenta un'area rettangolare dello schermo sulla quale l'applicazione può disegnare o recuperare eventi provocati dall'utente.
`VSPACE` (opzionale)	Definisce la spaziatura verticale
`PARAM` (opzionale)	Definisce i valori da passare all'Applet
`WIDTH` (non opzionale)	Definisce la larghezza del canvas dell'Applet che non può essere ridefinita a runtime

College
elis SCHOOL

Il metodo main nelle Applet

- L'Applet non deve obbligatoriamente definire il metodo main come in una normale applicazione Java perché il suo entry-point è costituito dal metodo init.
- Nonostante questo è utile scrivere il metodo main all'interno della Applet per almeno due ragioni:
  - permette di effettuare il test dell'Applet senza ricorrere all'AppletViewer o al browser;
  - la sua presenza permette il funzionamento del programma anche come applicazione standalone.
- Benché si utilizzino regole diverse per creare Applet e applicazioni, esse non entrano in conflitto tra loro, dato che il lifecycle delle Applet è ignorato quando il programma viene eseguito come applicazione.

Il metodo main nelle Applet (cont.)

- Nell'esempio seguente, l'Applet viene visualizzata all'interno di un oggetto di classe Frame nel caso di funzionamento come applicazione.

```
import java.awt.ActiveEvent;
import java.awt.event.WindowEvent;
import java.awt.Frame;
import java.awt.event.WindowAdapter;
import java.applet.Applet;
import java.awt.Button;
import java.awt.TextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import java.util.Random;
```

```
public class MyApplet extends Applet
    implements ActionListener{

  Button P1, P2;
  TextField testo;
  public void init() {
    P1=new Button("pari");
    add(P1);
    P1.addActionListener(this);
    P2=new Button("dispari");
    add(P2);
    P2.addActionListener(this);
    testo =new TextField();
    testo.setColumns(30);
    add(testo);
  }
```

Il metodo main nelle Applet (cont.)

```
public static void main(String[] args) {
    MyApplet applet = new MyApplet();

    Frame frame = new Frame("MyApplet");
    frame.addWindowListener(new
    WindowAdapter() {
      public void windowClosing(WindowEvent e)
    {
        System.exit(0);
      }
    });

  frame.add(applet);
  frame.setSize(200, 200);
  applet.init();
  frame.setVisible(true);
  }
```

```
public void actionPerformed(ActionEvent ae)
    {
    int i,a;
    String valore;
    Random r=new Random();
    a=Math.abs(r.nextInt())%2;
    valore=(String) ae.getActionCommand();
    if ((a==0)&&(valore.equals("pari")))
      testo.setText(" Hai vinto!");
    else if
    ((a==1)&&(valore.equals("dispari")))
     testo.setText(" Hai vinto!");
    else testo.setText(" Hai perso!");
  }
}
```

L’interfaccia grafica: da AWT a Swing

- La scelta fatta per il package AWT, cioè quella di voler creare una libreria grafica che fosse un massimo comune denominatore fra le librerie grafiche dei vari sistemi operativi, si prefiggeva di realizzare un ambiente grafico conforme a quello della piattaforma ospite.
- Il fatto di essere pioniera in tal senso è la causa prima che rende AWT una libreria piuttosto povera, con interfacce grafiche scarne e con un numero di funzionalità limitato.
- Per aumentare le potenzialità espressive del linguaggio, la progettazione della libreria Swing è stata affrontata ripensando al modello implementativo e rifacendosi al noto Model View Controller.
- L’interfaccia Swing risulta così leggera (*lightweight*), in grado di gestire un maggior numero di eventi e indipendente dall’implementazione delle classi grafiche del sistema operativo ospitante.

College
elis SCHOOL

L'interfaccia grafica: da AWT a Swing (cont.)

- Le Applet in swing non sono più estensione della classe Applet ma della JApplet che fa parte del package javax.swing. Per quel che riguarda la costruzione dell'interfaccia grafica, l'aggiunta di componenti non viene più fatta direttamente sul contenitore dell'Applet

  add(<componente grafico>);

  ma sul ContentPane, un pannello di contenimento accessibile attraverso il metodo getContentPane(), delegato a contenere i componenti grafici.

  this.getContentPane().contentPane.add(<componente grafico>);

  equivalente a

  Container contentPane = this.getContentPane();

  contentPane.add(<Componente grafico>);

College
elis SCHOOL

Esempio

```
<HTML>
   <HEAD>
   <TITLE>Un'applet di prova</TITLE >
   </HEAD >
   <BODY>
        <H1 ALIGN="center">applet- swing</H1>
        <P ALIGN="center">
        <APPLET
   CODE="it/elis/appletlesson/AppletWithParameterSwing
   .class" WIDTH=400 HEIGHT=300>
                <!-- Passaggio di parametri -->
                <PARAM NAME=message VALUE="Ecco il
   valore che l'Applet passa...">
                <PARAM NAME=number VALUE=15>
        </APPLET></P>
</BODY>
```


College

elis SCHOOL

Esempio (cont.)

```
1  package it.elis.appletlesson;
2
3  import javax.swing.JApplet;
4  import javax.swing.JLabel;
5  public class AppletWithParameterSwing extends JApplet {
6    private String message;
7    private String number;
8    private JLabel viewMessage;
9    private JLabel numText;
10   public void init() {
11     message = getParameter("message");
12     if(message == null)
13       message= "Manca il tag HTML message";
14     viewMessage = new JLabel(message);
15     number = getParameter("number");
16     if(number != null) {
17         try {
18           int num = Integer.parseInt(number);
19           numText = new JLabel(number);
20         }catch(NumberFormatException e) {
21           numText = new JLabel("Non è stato inserito un numero");
22         }
23       }else {
24         numText = new JLabel("Manca il tag HTML number");
25       }
26     //aggiunti componenti al Container di alto livello
27     this.getContentPane().add("Center", viewMessage);
28     this.getContentPane().add("South", numText);
29     }
30   }
```


College
eLIS SCHOOL

Euro convertitore

- Obiettivo: scrivere un'applet che permetta di fare le conversioni lire – euro e viceversa

College
elis SCHOOL

Euro convertitore 2 (componenti)

- Per prima cosa pensiamo alla parte grafica, poi passeremo alla gestione degli eventi
- Abbiamo bisogno di
  - Un'etichetta (con la scritta "Euro Convertitore")
  - Due caselle di testo e due etichette, una con la scritta "lire" e l'altra con la scritta "euro"

College
elis SCHOOL

Euro convertitore 3

```
package it.elis.appletlesson;

/*<applet code="euroConvertitore.class" width="380"
height="100">
</applet>*/

import java.awt.*;
import java.applet.*;
public class euroConvertitore extends Applet {
public void init() {
this.add(new Label("Euro Convertitore"));
TextField lire = new TextField(10);
TextField euro = new TextField(10);
this.add(new Label("Lire"));
this.add(lire);
this.add(new Label("Euro"));
this.add(euro);
}
}
```


College

elis SCHOOL

Euro convertitore 4 (layout)

- Per aggiustare il layout impostiamo il gestore a BorderLayout
- Aggiungiamo a north un Panel (che verrà "stirato") e a questo panel aggiungiamo la Label "Euro Convertitore
- Aggiungiamo un altro Panel al centro e poi a questo le restanti due etichette e caselle di testo

College
elis SCHOOL

Euro convertitore 5

```
..
public void init() {
this.setLayout(new BorderLayout());
Panel p = new Panel();
Panel p2 = new Panel();
p2.add(new Label("Euro Convertitore"));
this.add(p2, BorderLayout.NORTH);
this.add(p, BorderLayout.CENTER);
TextField lire = new TextField(10);
TextField euro = new TextField(10);
p.add(new Label("Lire"));
p.add(lire);
p.add(new Label("Euro"));
p.add(euro);
}
}
```


College

elis SCHOOL

Euro convertitore 6 (eventi)

- Vogliamo che quando si scrive nella casella di testo delle lire automaticamente venga aggiornata la casella degli euro
- L'evento da gestire è il cambiamento del testo e quindi dobbiamo scrivere un TextListener

College
elis SCHOOL

Euro convertitore 7

```
class convertitore implements TextListener {
        double tasso;
        TextComponent out;
        convertitore(TextComponent output, double tasso){
                this.out = output;
                this.tasso = tasso; }
        public void textValueChanged(TextEvent e) {
                TextComponent c = (TextComponent)e.getSource();
                double d = 0; String s = "";
                d = Double.parseDouble(c.getText());
                s += (d*tasso);
                out.setText(s); }
}
```


College

elis SCHOOL

Euro convertitore 8

```
public class euroConvertitore extends Applet {
        public void init() {
    …
        TextField lire = new TextField(10);
        TextField euro = new TextField(10);
        convertitore l2e = new convertitore(euro,(1/1936.27));
        convertitore e2l = new convertitore(lire,1936.27);
        lire.addTextListener(l2e);
        euro.addTextListener(e2l);
    …
}
}
```


College

elis SCHOOL

Euro convertitore 9 (problemi)

- Non abbiamo tenuto conto di
  - Errori di calcolo ed arrotondamenti
  - Errori inseriti dall'utente
  - Ogni volta che si cambia il testo in una casella viene lanciato un TextEvent che viene catturato da un TextListener il quale a sua volta cambia il testo nell'altra casella che, a sua volta, lancerà un altro TextEvent e così via in un loop infinito

College
elis SCHOOL

Euro convertitore 10 (loop)

- Per risolvere il problema del loop di eventi facciamo in modo che la classe ascoltatore cambi il testo della casella di output, solo se oltre ad avere il testo modificato ha anche il focus

College
elis SCHOOL

Euro convertitore 11

```
class convertitore implements TextListener,
FocusListener {
      double tasso;
      TextComponent out;
      boolean editing = false;
      …
      public void textValueChanged(TextEvent e) {
      …
            if (editing)
            out.setText(s);
      }
      public void focusGained(FocusEvent e) {
            editing = true; }
      public void focusLost(FocusEvent e) {
            editing = false; } }
```

Euro convertitore 12

```
public class euroConvertitore extends
Applet {
     public void init() {
     …
          lire.addTextListener(l2e);
          euro.addTextListener(e2l);
          lire.addFocusListener(l2e);
          euro.addFocusListener(e2l);
     …
          }
     }
```


College

elis SCHOOL

Euro convertitore 13 (errori)

- Se l'utente sbaglia ad inserire i dati il metodo Double.parseDouble lancia un'eccezione NumberFormatException. Catturiamo l'eccezione e scriviamo "errore" nella casella di testo
- Infine scriviamo una funzione per arrotondare ad un certo numero di cifre decimali

Euro convertitore 14

```
class convertitore implements TextListener, FocusListener {
            …
            public void textValueChanged(TextEvent e) {
                        …
                        double d = 0; String s = "";
                        try {
                                    d = Double.parseDouble(c.getText());
                                    s += (this.arrotonda(d*tasso,16));
                                    }
                        catch(NumberFormatException ex) {
                                    s = "errore!";}
                                    … }
                        …
            double arrotonda(double d, int p) {
                        return Math.rint(d*Math.pow(10,p))/Math.pow(10,p);
            }
}
```

Altro esempio

- Scriveremo un applet che:
  - Tramite il **metodo init** carichi **una immagine**
  - Tramite il **metodo paint** la **disegni sullo schermo**
  - Utilizzando **mouseMove** faccia **rincorrere il mouse** dall'immagine

Altro esempio (2)

```
import java.awt.*;          // Le classi per il disegno
import java.applet.Applet; // La superclasse

public class mouse extends Applet {

private final int MAXX = 250; // Massima estensione in X (verso destra)
private final int MAXY = 150; // Massima estensione in Y (verso il basso)
private int xo = 0;           // Coordinate correnti dell'immagine
private int yo = 0;
private Image dot = null;     // L'immagine da muovere sullo schermo

  public void init () {                          // Metodo chiamato all'avvio dell'applet
    dot = getImage(getCodeBase(), "lista.gif"); // Lancia il caricamento dell'immagine
  }

  public boolean mouseMove (Event evt, int x, int y) { // Risponde al mouse
    xo = (xo + x) / 2;          // La nuova x e' la media tra la posizione attuale e quella del mouse
    if (xo > MAXX) xo = MAXX;  // Se supera il massimo, la fermo al massino
    yo = (yo + y) / 2;          // La nuova y e' la media tra la posizione attuale e quella del mouse
    if (yo > MAXY) yo = MAXY;  // Se supera il massimo, la fermo al massino
    repaint ();                 // Ordino di ridisegnare l'area dell'Applet
    return true;                // Ritorno 'true' per dire che ho gestito l'evento
  }

  synchronized public void paint(Graphics g) { // Metodo invocato per ridisegnare l'Applet
    g.drawImage (dot, xo, yo, this);           // Disegna l'immagine alle coordinate memorizzate
  }
}
```